Non determinism through type isomorphism

Alejandro Díaz-Caro*
Université Paris 13, Sorbonne Paris Cité, LIPN
Université Paris-Ouest Nanterre La Défense
INRIA

Gilles Dowek
INRIA
23 avenue d'Italie, CS 81321,
75214 Paris Cedex 13

We define an equivalence relation on propositions and a proof system where equivalent propositions have the same proofs. The system obtained this way resembles several known non-deterministic and algebraic lambda-calculi.

1 Introduction

Several non-deterministic extensions to the λ-calculus have been proposed, e.g. [6, 7, 10–12, 24]. In these approaches, the parallel composition (sometimes called the *must-convergent* parallel composition) is such that if $\mathbf{r}$ and $\mathbf{s}$ are two λ-terms, the term $\mathbf{r}+\mathbf{s}$ (also written $\mathbf{r} \parallel \mathbf{s}$) represents the computation that runs either $\mathbf{r}$ or $\mathbf{s}$ non-deterministically. It is common to consider in these approaches the associativity and commutativity of the operator $+$. Indeed the interpretation "either $\mathbf{r}$ or $\mathbf{s}$ runs" shall not prioritise any of them, and so "either $\mathbf{s}$ or $\mathbf{r}$ runs" must be represented by the same term. Moreover, $(\mathbf{r}+\mathbf{s})\mathbf{t}$ can run either $\mathbf{rt}$ or $\mathbf{st}$, which is the same expressed by $\mathbf{rt}+\mathbf{st}$. Extra equivalences (or rewrite rules, depending on the presentation) are set up to account for such an interpretation, e.g. $(\mathbf{r}+\mathbf{s})\mathbf{t} \leftrightarrow \mathbf{rt}+\mathbf{st}$. This right distributivity can alternatively be seen as the one of function sum: $(\mathbf{f}+\mathbf{g})(x)$ is defined pointwise as $\mathbf{f}(x)+\mathbf{g}(x)$. This is the approach of the algebraic lambda-calculi [3, 26], two independently introduced algebraic extensions which resulted strongly related afterwards [4, 15]. In these algebraic calculi, a scalar pondering each 'choice' is considered in addition to the sum of terms.

Because of these equivalences between terms, it is natural to think that a typed version must allow some equivalences at the type level. Definitely, if $\mathbf{r}$ and $\mathbf{s}$ are typed with types A and B respectively, it is natural to expect that whatever connective tie these types in order to type $\mathbf{r}+\mathbf{s}$, it must be commutative and associative.

An independent stream of research is the study of isomorphisms between types for several languages (see [13] for a reference). For example, we know that the propositions $A \wedge B$ and $B \wedge A$ are equiprovable: one is provable if and only if the other is, but they do not have the same proofs. If $\mathbf{r}$ is a proof of A and $\mathbf{s}$ is a proof of B, then $\langle \mathbf{r}, \mathbf{s} \rangle$ is a proof of $A \wedge B$ while $\langle \mathbf{s}, \mathbf{r} \rangle$ is a proof of $B \wedge A$. Despite that both proofs can be derived from the same hypotheses, they are not the same. In this paper, we show how the non-determinism arises naturally in a classic context only by introducing some equivalences between types. These equivalences, nevertheless, will be chosen among valid, well-known isomorphisms. In order to consider these isomorphic types as equivalent, we need to design a proof system such that they have the same proofs, or conversely, in order to consider these terms to be equivalent, we need to make these isomorphic types to be equivalent. Formally, two types A and B are isomorphic if there are two conversion functions f of type $A \Rightarrow B$ and g of type $B \Rightarrow A$, such that $g(f(x)) = x$ for any x of type A and $f(g(y)) = y$ for any y of type B. Hence, in this system the conversion functions f and g should become and identity function. In other words, we take the quotient of the set of propositions by the relation

*This work was supported by grants from DIGITEO and Région Île-de-France.

D. Kesner and P. Viana (Eds.): LSFA 2012
EPTCS 113, 2013, pp. 137–144, doi:10.4204/EPTCS.113.13

generated by the isomorphisms of types and define proofs for elements in this quotient. In System F with products, which correspond to the propositional logic with universal quantifier, conjunction and implication, the full list of isomorphisms is known [13], and it is summarised in Figure 1.

1. $A \wedge B \equiv B \wedge A$	6. $\forall X.\forall Y.A \equiv \forall Y.\forall X.A$
2. $A \wedge (B \wedge C) \equiv (A \wedge B) \wedge C$	7. $\forall X.A \equiv \forall Y.A[Y/X]$
3. $A \Rightarrow (B \wedge C) \equiv (A \Rightarrow B) \wedge (A \Rightarrow C)$	8. $\forall X.(A \Rightarrow B) \equiv A \Rightarrow \forall X.B$ if $X \notin FV(A)$
4. $(A \wedge B) \Rightarrow C \equiv A \Rightarrow (B \Rightarrow C)$	9. $\forall X.(A \wedge B) \equiv \forall X.A \wedge \forall X.B$
5. $A \Rightarrow (B \Rightarrow C) \equiv B \Rightarrow (A \Rightarrow C)$	10. $\forall X.(A \wedge B) \equiv \forall X.\forall Y.(A \wedge (B[Y/X]))$

Figure 1: All the type isomorphisms in propositional logic with universal quantifier, non-idempotent conjunction and implication

In this work, we consider only the three first isomorphisms of this list, because they are those that arise naturally when studying non deterministic processes. The impact of the others is left for future work.

Usually, for the deduction rule on the right if we call $\mathbf{r}$ the proof of A and $\mathbf{s}$ that of B, we write $\mathbf{r},\mathbf{s}$ or $\langle \mathbf{r},\mathbf{s}\rangle$ the proof of $A \wedge B$. However if $A \wedge B$ and $B \wedge A$ are *the same* proposition, we get $\mathbf{r},\mathbf{s}$ and $\mathbf{s},\mathbf{r}$ to be the same term. Let us write "$+$" to the commutative comma[1] and set the rule

$$\frac{A \qquad B}{A \wedge B}$$

$$\frac{\mathbf{r} : A \qquad \mathbf{s} : B}{\mathbf{r} + \mathbf{s} : A \wedge B}.$$

In the same way, the associativity of $\wedge$ induces that of $+$. Furthermore, the isomorphism (3) of Figure 1 induces the following equivalence on proofs. If $\mathbf{r}$ is a proof of $A \Rightarrow B$, $\mathbf{s}$ one of $A \Rightarrow C$, and $\mathbf{t}$ one of A then $\mathbf{r}+\mathbf{s}$ is a proof of $A \Rightarrow (B \wedge C)$ and $(\mathbf{r}+\mathbf{s})\mathbf{t}$ is a proof of $B \wedge C$. This proof is the same as $\mathbf{rt}+\mathbf{st}$. Summarising, from the equivalences between types we obtained a commutative and associative $+$, which is such that the application right-distributes over it.

Several non-classical type systems have been already proposed for the non-deterministic and algebraic calculi, e.g. [1, 2, 16]. In these systems there is already an equivalence relation on propositions such that if $A \equiv B$ and A types a term, then also B types it. Such equivalence is reminiscent of type theory [9, 22] and deduction modulo [17, 19]. But here we go further, introducing an equivalence relation that equates types built with different connectives such as $A \Rightarrow (B \wedge C)$ and $(A \Rightarrow B) \wedge (A \Rightarrow C)$, which is not possible there. Moreover, there is no elimination rule for conjunction in [1, 2, 16]. Indeed, having commutativity and associativity properties in both, the sums of terms and the conjunctions of propositions, leads to uncertainty on how to eliminate them. A rule like "$\mathbf{r} : A \wedge B$ implies $\pi_1(\mathbf{r}) : A$", would not be consistent. If A and B are two arbitrary types, $\mathbf{s}$ a term of type A and $\mathbf{t}$ a term of type B, then $\mathbf{s}+\mathbf{t}$ has both types $A \wedge B$ and $B \wedge A$, thus $\pi_1(\mathbf{s}+\mathbf{t})$ would have both type A and type B. Hence, a naive rule would lead to inconsistency. The projection would project a random term of any of the types of its arguments, so not being a trustfully valid proof for any proposition.

The approach we follow here is to consider explicitly typed terms (Church style), and hence make the projection to depend on the type: if $\mathbf{r} : A \wedge B$ then $\pi_A(\mathbf{r}) : A$. This way, we recover consistency of the proof system. This new form of projection entails allowing some non-determinism directly in the rewrite system. Indeed, if $\mathbf{r}$ and $\mathbf{s}$ have the same type A, $\pi_A(\mathbf{r}+\mathbf{s})$ both reduces to $\mathbf{r}$ and to $\mathbf{s}$. A priori

[1] We could chose another symbol, however $+$ is the one used in most non-deterministic settings.

this does not entail any problem; any of them is a valid proof of the same proposition A. This approach can be summarised by the slogan *"the subject reduction property is more important than the uniqueness of results"* [18]. Therefore the projection turns the non-deterministic choice explicit.

We formalise all of the previously discussed concepts in Section 2, where we present the calculus λ_+, and provide some examples. Section 3 The next section is devoted to prove that our system enjoys the subject reduction property. In Section 4 we discuss the relation of this setting with respect to the algebraic approach. Finally, Section 5 concludes the paper with suggestions for future research.

2 The calculus

2.1 Definitions

In this section we present the calculus λ_+, an explicitly typed lambda-calculus extended with a $+$ operator as discussed in the introduction. We consider the following grammar of types

$$A,B,C,\ldots ::= \quad X \mid A \Rightarrow B \mid A \wedge B \mid \forall X.A \ ,$$

where the isomorphisms (1), (2) and (3) from Figure 1 are made explicit by an equivalence relation between types

$$A \wedge B \equiv B \wedge A \quad , \quad (A \wedge B) \wedge C \equiv A \wedge (B \wedge C) \quad \text{and} \quad A \Rightarrow (B \wedge C) \equiv (A \Rightarrow B) \wedge (A \Rightarrow C) \ .$$

The set of terms Λ is defined inductively by the grammar

$$\mathbf{r},\mathbf{s},\mathbf{t} ::= \quad x^A \mid \lambda x^A.\mathbf{r} \mid \mathbf{rs} \mid \mathbf{r}+\mathbf{s} \mid \pi_A(\mathbf{r}) \mid \Lambda X.\mathbf{r} \mid \mathbf{r}\{A\} \ .$$

All our variable occurrences are explicitly typed, but we usually omit the superscript indicating the type of variables when it is clear from the context. For example we write $\lambda x^A.x$ instead of $\lambda x^A.x^A$. The α*-conversion* and the sets $FV(\mathbf{r})$ of *free variables of* $\mathbf{r}$ and $FV(A)$ of *free variables of* A are defined as usual in the λ-calculus (cf. [5, §2.1]). For example $FV(x^A y^B) = \{x^A, y^B\}$. The same variable, with different types, is treated as a different variable. For example, the term $\lambda x^A.x^B : A \Rightarrow B$ is typable in our system, and it is the constant function x^B, since x^B is free in the term $\lambda x^A.x^B$. We say that a term $\mathbf{r}$ is *closed* whenever $FV(\mathbf{r}) = \emptyset$. Given two terms $\mathbf{r}$ and $\mathbf{s}$ we denote by $\mathbf{r}[\mathbf{s}/x]$ the term obtained by simultaneously substituting the term $\mathbf{s}$ for all the free occurrences of x in $\mathbf{r}$, subject to the usual proviso about renaming bound variables in $\mathbf{r}$ to avoid capture of the variables free in $\mathbf{s}$. Analogously $A[B/X]$ denotes the substitution of the type B for all the free occurrences of X in A, and $\mathbf{r}[B/X]$ the substitution in $\mathbf{r}$. For example, $(x^A)[B/Y] = x^{(A[B/Y])}$, $(\lambda x^A.\mathbf{r})[B/X] = \lambda x^{A[B/X]}.\mathbf{r}[B/X]$ and $(\pi_A(\mathbf{r}))[B/X] = \pi_{A[B/X]}(\mathbf{r}[B/X])$. Simultaneous substitutions are defined in the same way. Finally, terms and types are considered up to α-conversion.

Each term of the language has a main type associated, which can be obtained from the type annotations, and other types induced by the type equivalences. The type system for λ_+ is given in Figure 2. If $FV(\mathbf{r}) = \{x_1^{A_1},\ldots,x_n^{A_n}\}$, we write $\Gamma(\mathbf{r}) = \{A_1,\ldots,A_n\}$. $FV(\{A_1,\ldots,A_n\})$ is defined by $\bigcup_{i=1}^n FV(A_i)$. *Typing judgements* are of the form $\mathbf{r} : A$. A term $\mathbf{r}$ is *typable* if there exists a type A such that $\mathbf{r} : A$.

Lemma 2.1 states that the typing modulo equivalences is unique.

Lemma 2.1. *If* $\mathbf{r} : A$ *and* $\mathbf{r} : B$*, then* $A \equiv B$*.*

Proof. Without rule $\equiv$, the type system is syntax directed. The only rule able to modify the type of a term without changing it is $\equiv$. □

$$\frac{}{x^A : A}\, ax \qquad [A \equiv B]\frac{\mathbf{r} : A}{\mathbf{r} : B} \equiv \qquad \frac{\mathbf{r} : B}{\lambda x^A.\mathbf{r} : A \Rightarrow B} \Rightarrow_I \qquad \frac{\mathbf{r} : A \Rightarrow B \quad \mathbf{s} : A}{\mathbf{rs} : B} \Rightarrow_E$$

$$\frac{\mathbf{r} : A \quad \mathbf{s} : B}{\mathbf{r}+\mathbf{s} : A \wedge B} \wedge_I \qquad \frac{\mathbf{r} : A \wedge B}{\pi_A(\mathbf{r}) : A} \wedge_E \qquad [X \notin FV(\Gamma(\mathbf{r}))]\frac{\mathbf{r} : A}{\Lambda X.\mathbf{r} : \forall X.A} \forall_I \qquad \frac{\mathbf{r} : \forall X.A}{\mathbf{r}\{B\} : A[B/X]} \forall_E$$

Figure 2: The type system for λ_+

The operational semantics of the calculus is given in Figure 3, where there are two distinct relations between terms: $\hookrightarrow$ and a symmetric relation $\rightleftarrows$. We write $\rightleftarrows^*$ and $\hookrightarrow^*$ for the transitive and reflexive closures of $\rightleftarrows$ and $\hookrightarrow$ respectively. In particular, notice that $\rightleftarrows^*$ is an equivalence relation.

Symmetric relation:

$\mathbf{r}+\mathbf{s} \rightleftarrows \mathbf{s}+\mathbf{r}$, $\quad (\mathbf{r}+\mathbf{s})+\mathbf{t} \rightleftarrows \mathbf{r}+(\mathbf{s}+\mathbf{t})$, $\quad (\mathbf{r}+\mathbf{s})\mathbf{t} \rightleftarrows \mathbf{rt}+\mathbf{st}$,
$\lambda x^A.(\mathbf{r}+\mathbf{s}) \rightleftarrows \lambda x^A.\mathbf{r}+\lambda x^A.\mathbf{s}$, $\quad$ If $\mathbf{r} : A \Rightarrow (B \wedge C)$, then $\pi_{A\Rightarrow B}(\mathbf{r})\mathbf{s} \rightleftarrows \pi_B(\mathbf{rs})$.

Reductions:

$(\lambda x^A.\mathbf{r})\ \mathbf{s} \hookrightarrow \mathbf{r}[\mathbf{s}/x]$, $\quad (\Lambda X.\mathbf{r})\{A\} \hookrightarrow \mathbf{r}[A/X]$, $\quad$ If $\mathbf{r} : A$, then $\pi_A(\mathbf{r}+\mathbf{s}) \hookrightarrow \mathbf{r}$.

Figure 3: Operational semantics of λ_+

2.2 Examples

Example 2.2. We have $\lambda x^{A\wedge B}.x : (A \wedge B) \Rightarrow (A \wedge B)$ and so by rule $\equiv$, $\lambda x^{A\wedge B}.x : ((A \wedge B) \Rightarrow A) \wedge ((A \wedge B) \Rightarrow B)$, from which we can obtain $\pi_{(A\wedge B)\Rightarrow A}(\lambda x^{A\wedge B}.x) : (A \wedge B) \Rightarrow A$. Let $\mathbf{r} : A \wedge B$, then $\pi_{(A\wedge B)\Rightarrow A}(\lambda x^{A\wedge B}.x)\mathbf{r} : A$, and notice that $\pi_{(A\wedge B)\Rightarrow A}(\lambda x^{A\wedge B}.x)\mathbf{r} \rightleftarrows \pi_A((\lambda x^{A\wedge B}.x)\mathbf{r}) \hookrightarrow \pi_A(\mathbf{r})$.

Example 2.3. Let $\mathbf{TF} = \lambda x^A.\lambda y^B.(x+y)$. It is easy to check that $\mathbf{TF} : A \Rightarrow B \Rightarrow (A \wedge B)$, and by rule $\equiv$ it also has the type $(A \Rightarrow B \Rightarrow A) \wedge (A \Rightarrow B \Rightarrow B)$. Therefore, $\pi_{A\Rightarrow B\Rightarrow A}(\mathbf{TF}) : A \Rightarrow B \Rightarrow A$ is well typed. In addition, if $\mathbf{r} : A$ and $\mathbf{s} : B$, we have $\pi_{A\Rightarrow B\Rightarrow A}(\mathbf{TF})\mathbf{rs} : A$.

Notice that $\pi_{A\Rightarrow B\Rightarrow A}(\mathbf{TF})\mathbf{rs} \rightleftarrows \pi_{B\Rightarrow A}(\mathbf{TFr})\mathbf{s} \rightleftarrows \pi_A(\mathbf{TFrs}) \hookrightarrow \pi_A((\lambda y^B.(\mathbf{r}+y))\mathbf{s}) \hookrightarrow \pi_A(\mathbf{r}+\mathbf{s}) \hookrightarrow \mathbf{r}$, which is coherent with such typing.

Example 2.4. Let $\mathbf{T} = \lambda x^A.\lambda y^B.x$ and $\mathbf{F} = \lambda x^A.\lambda y^B.y$. Then $\mathbf{T}+\mathbf{F} : (A \Rightarrow B \Rightarrow A) \wedge (A \Rightarrow B \Rightarrow B)$, hence $\pi_{(A\Rightarrow B\Rightarrow A)\wedge(A\Rightarrow B\Rightarrow B)}(\mathbf{T}+\mathbf{F}+\mathbf{TF})$ reduces non-deterministically either to $\mathbf{T}+\mathbf{F}$ or to $\mathbf{TF}$. Moreover, notice that $\mathbf{T}+\mathbf{F} \rightleftarrows^* \mathbf{TF}$, hence in this very particular case, the non-deterministic choice does not play any role.

3 Subject reduction

In this section we prove that the set of types assigned to a term is invariant under $\rightleftarrows$ and $\hookrightarrow$. In other words, Theorem 3.2 states that if $\mathbf{r}$ is a proof of A, any reduction fired from $\mathbf{r}$ will still be a proof of A.

The substitution lemma below will be the key ingredient in the proof of subject reduction. It ensures that when substituting types for type variables or terms for term variables, in an adequate manner, the typing judgements remain valid.

Lemma 3.1 (Substitution)**.** *If $\mathbf{r} : B$ and $\mathbf{s} : A$, then $\mathbf{r}[\mathbf{s}/x^A] : B$. Also, If $\mathbf{r} : A$, then $\mathbf{r}[B/X] : A[B/X]$.*

Proof. By induction over $\mathbf{r}$ for the first result and over the type derivation for the second. □

Now we can prove the subject reduction property, ensuring that the typing is preserved during reduction.

Theorem 3.2 (Subject reduction)**.** *If $\mathbf{r} \to \mathbf{s}$ and $\mathbf{r} : A$, then $\mathbf{s} : A$ (where $\to$ is either $\hookrightarrow$ or $\rightleftarrows$).*

Proof. By induction over the reduction relation. We give only two interesting cases.
Rule $\pi_{A\Rightarrow B}(\mathbf{r})\mathbf{s} \rightleftarrows \pi_B(\mathbf{rs})$, with $\mathbf{r} : A \Rightarrow (B \wedge C)$. Let $\pi_{A\Rightarrow B}(\mathbf{r})\mathbf{s} : D$, then $\pi_{A\Rightarrow B}(\mathbf{r}) : E \Rightarrow D$ and $\mathbf{s} : E$. But then $E \equiv A$ and $D \equiv B$, because clearly, the main type for $\pi_{A\Rightarrow B}(\cdot)$ is $A \Rightarrow B$, so $\mathbf{r} : (A \Rightarrow B) \wedge F$, however since $\mathbf{r} : A \Rightarrow (B \wedge C)$, we have $F \equiv A \Rightarrow C$. So, by rule $\Rightarrow_E$, $\mathbf{rs} : B \wedge C$. We conclude by rule $\wedge_E$. For the inverse direction, let $\pi_B(\mathbf{rs}) : D$. Then $D \equiv B$ and $\mathbf{rs} : B \wedge E$, so $\mathbf{r} : F \Rightarrow (B \wedge E)$ and $\mathbf{s} : F$. Hence, since $\mathbf{r} : A \Rightarrow (B \wedge C)$, by Lemma 2.1, we have $F \equiv A$ and $E \equiv C$, so $\pi_{A\Rightarrow B}(\mathbf{r}) : A \Rightarrow B$, from which, we conclude $\pi_{A\Rightarrow B}(\mathbf{r})\mathbf{s} : B$. We conclude by rule $\equiv$.
Rule $(\lambda x^A.\mathbf{r})\ \mathbf{s} \hookrightarrow \mathbf{r}[\mathbf{s}/x]$. Let $(\lambda x^A.\mathbf{r})\mathbf{s} : B$, then $\lambda x^A.\mathbf{r} : C \Rightarrow D$ and $\mathbf{s} : C$, with $D \equiv B$. Then $\mathbf{r} : E$, with $A \Rightarrow E \equiv C \Rightarrow D$. Notice that, since $A \Rightarrow E \equiv C \Rightarrow D$, it must be $A \equiv C$ and $E \equiv D$. Then, by rule $\equiv$, $\mathbf{s} : A$, and so, by Lemma 3.1, $\mathbf{r}[\mathbf{s}/x^A] : E$, and since $E \equiv D \equiv B$, by rule $\equiv$, we obtain $\mathbf{r}[\mathbf{s}/x^A] : B$. □

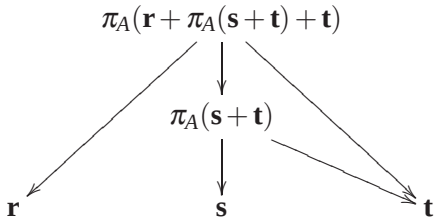
4 From non-determinism to probabilities

In [3] and [26] two algebraic extensions of the untyped lambda-calculus are introduced, which we call λ_{lin} and λ_{alg} respectively. In these settings, not only the $+$ operator is present, but also a scalar pondering each choice. Hence, if $\mathbf{r}$ and $\mathbf{s}$ are two possible terms, so is the linear combination of them $\alpha.\mathbf{r} + \beta.\mathbf{s}$, with α, β some kind of scalars (taken from a generic ring in λ_{lin} or from $\mathbb{R}^{\geq 0}$ in λ_{alg}). Both these calculi identify the term $(\mathbf{r} + \mathbf{s})\mathbf{t}$ with $\mathbf{rt} + \mathbf{st}$, either with a rewrite system or an equality, and $+$ is associative and commutative. Also, the scalars interact with the $+$, e.g. $\mathbf{r} + \mathbf{r} \leftrightarrow 2.\mathbf{r}$. By restricting the scalars to positive real numbers, or even to natural numbers, one possible interpretation is that the scalars give the probability of following one possible path (after 'normalising' the scalars, i.e. dividing over the total amount in order to sum up to 1). In this way, the term $2.\mathbf{r} + \mathbf{s}$ is twice more likely to run $\mathbf{r}$ than $\mathbf{s}$.

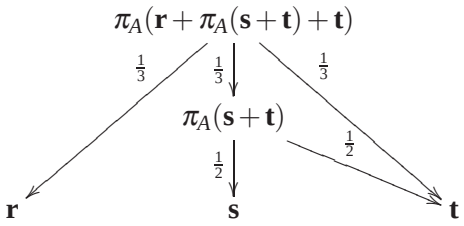
Indeed, in [1, §6] the type system B for λ_{lin} is proposed, which can decide whether a superposition is a probability distribution (i.e. it can check that the sum of terms is up to 1). Such a system includes scalars at the type level, reflecting those in the terms, so $\alpha.\mathbf{r}$ has type $\alpha.A$ whenever $\mathbf{r}$ has type A. This provides a powerful tool to account for the scalars within the terms, however it entails a 'non-classical' extension of System F with scalars pondering the types. In such a formalism, there is no possibility to tie terms with different types: if $\mathbf{r}$ and $\mathbf{s}$ have both type A, then $\alpha.\mathbf{r} + \beta.\mathbf{s}$ have type $(\alpha + \beta).A$, however if the types of $\mathbf{r}$ and $\mathbf{s}$ differ, the previous term cannot be typed. That weakness is solved in [2], where a more powerful system is introduced, with a type system also allowing for linear combination of types, just like for terms. In both these systems, while powerful, it is hard to establish a connection with a well-known logic. That is precisely the goal of [8], where a more 'classic' system is developed, with no scalars at the type level. However it carries some costs: first, it is only meant for positive real scalars (which anyway is enough for a 'probabilistic' interpretation), and more importantly, the type system gives just an approximation, an upper bound, of the scalars in the terms.

We could envisage extending λ_+ with a more thorough projection where $\pi_A(\alpha.\mathbf{r} + \beta.\mathbf{s})$ would output either $\mathbf{r}$, with probability α, or $\mathbf{s}$ with probability β. However, even when the scalars are not explicitly written, the probabilities are present. The following example is clarifying.

Let $\mathbf{r} : A$, $\mathbf{s} : A$ and $\mathbf{t} : A$. Then, the reductions depicted in the diagram at right are possible. If we consider π_A making an equiprobable choice instead of a non-deterministic one, it is



clear that **t** have more probability to be reached, followed by **r**,
and the less likely is **s**.



Indeed, we can calculate the global probability of reaching each possibility by labelling the reductions with its local probability as shown in the diagram at left, from where just by summing up the labels reaching a term, and multiplying those in the same path, we can easily check that the term **r** has probability $\frac{1}{3}$ of being reached, the term **s** probability $\frac{1}{6}$ and the term **t** probability $\frac{1}{2}$. Hence, this term would be expressed with scalars as $\frac{1}{3}\mathbf{r}+\frac{1}{6}\mathbf{s}+\frac{1}{2}\mathbf{t}$ according to the previously discussed interpretation. Therefore, λ_+ could be seen as a sort of algebraic calculus, with implicit scalars taken from $\mathbb{Q}^{[0,1]}$, typed with a standard type system. These ideas will be fully developed in a future research.

5 Conclusions and future work

5.1 Conclusions

In this paper we have introduced λ_+, a proof system for second order propositional logic with an associative and commutative conjunction, and implication. In this system, isomorphic propositions get the same proofs. At this first step we only consider three isomorphisms, namely commutativity and associativity of the conjunction, and distributivity of implication with respect to conjunction. We use the symbol $+$ to put together the proofs of different propositions, so $\mathbf{r}+\mathbf{s}$ becomes a proof of $A\wedge B$, if **r** is a proof of A and **s** a proof of B. Such a symbol is commutative and associative, and application is right-distributive with respect to it, to account for the isomorphisms of propositions.

This construction entails a non-deterministic projection where if a proposition has two possible proofs, the projection of its conjunction can output any of them. For example, if **r** and **s** are two possible proofs of A, then $\pi_A(\mathbf{r}+\mathbf{s})$ will output either **r** or **s**.

In several works (cf. [21, §3.4] for a reference), the non-determinism is modelled by two operators. The first is normally written $+$, and instead of distributing over application, it actually makes the non-deterministic choice. Hence $(\mathbf{r}+\mathbf{s})\mathbf{t}$ reduces either to **rt** or to **st** [10]. The second one, denoted by $\|$, does not make the choice, and therefore $(\mathbf{r}\parallel\mathbf{s})\mathbf{t}$ reduces to $\mathbf{rt}\parallel\mathbf{st}$ [12]. One way to interpret these operators is that the first one is a non-deterministic one, while the second is the parallel composition. Another common interpretation is that $+$ is a *may-convergent* non-deterministic operator, where type systems ensure that at least one branch converges, while $\|$ is a *must-convergent* non-deterministic operator, where both branches are meant to converge [7, 14]. In our setting, the $+$ operator in λ_+ behaves like $\|$, and an extra operator (π_A) induces the non-deterministic choice. The main point is that this construction arose naturally just by considering some of the isomorphisms between types as an equivalence relation. In order to ensure that our system is must-convergent, we shall prove its strong normalisation, which is left for future research.

5.2 Open questions and future research

As mentioned in Section 4, the calculus λ_+ has implicit scalars on it, which can convert this non-deterministic setting into a probabilistic one. The original motivation behind λ_{lin} [3] and its *vectorial* type system [2] was to encode quantum computing on it. A projection depending on scalars could lead to

a measurement operator in a future design—after other questions like deciding orthogonality [25] have been addressed in that setting. This is a promising future direction we are willing to take.

In order to follow such direction, a first step is to move to a call-by-value calculus, where $\mathbf{r}(\mathbf{s}+\mathbf{t}) \not\rightleftarrows \mathbf{rs}+\mathbf{rt}$ (because a non-deterministic choice yet to make, is not considered to be a value). The reason to move to call-by-value is explained with the following example. Consider for instance the term $\delta = \lambda x.xx$ applied to a sum $\mathbf{r}+\mathbf{s}$. In call-by-name it reduces to $(\mathbf{r}+\mathbf{s})(\mathbf{r}+\mathbf{s})$ while in a call-by-value strategy (λ_{lin}) the same term reduces to $\delta\mathbf{r}+\delta\mathbf{s}$ first, and then to $\mathbf{rr}+\mathbf{ss}$. If seeking for a quantum interpretation, reducing $\delta(\mathbf{r}+\mathbf{s})$ into $(\mathbf{r}+\mathbf{s})(\mathbf{r}+\mathbf{s})$ is considered as the forbidden quantum operation of "cloning" [27], while the alternative reduction to $\mathbf{rr}+\mathbf{ss}$ is seen as a "copy", or CNOT, a fundamental quantum operation [23].

In order to account for such an equivalence, $\mathbf{r}(\mathbf{s}+\mathbf{t}) \not\rightleftarrows \mathbf{rs}+\mathbf{rs}$, we would need an equivalence at the type level such as $(A \wedge B) \Rightarrow C \equiv (A \Rightarrow C) \wedge (B \Rightarrow C)$, however it is clearly false. A workaround which have been used already in the vectorial type system [2] is to use the polymorphism instead of an equivalence. If $\mathbf{r}$ have type $\forall X.X \Rightarrow C_X$, then we can specialise X to the needed argument. Indeed, $\forall X.X \Rightarrow C_X$ entails both $A \Rightarrow C_A$ and $B \Rightarrow C_B$, which can latter be tied by a conjunction.

Another prominent future work is to determine what is needed for the remaining isomorphisms (cf. Figure 1). In a work by Garrigue and Aït-Kaci [20], the isomorphism $A \wedge B \equiv B \wedge A$ has been indirectly treated by combining it with currying: $(A \wedge B) \Rightarrow C \equiv A \Rightarrow B \Rightarrow C$ (cf. isomorphism (4) of Figure 1), from which it can be deduced the isomorphism $A \Rightarrow (B \Rightarrow C) \equiv B \Rightarrow (A \Rightarrow C)$ (cf. isomorphism (5) of Figure 1). Their proposal is the selective λ-calculus, a calculus including labellings to identify which argument is being used at each time. Moreover, by considering the Church encoding of pairs, isomorphism (5) implies isomorphism (1) (commutativity of $\wedge$). However their proposal is completely different to ours, and the non-determinism cannot be inferred from the selective λ-calculus.

Acknowledgements. We would like to thank Frédéric Blanqui, Michele Pagani and Giulio Manzonetto for enlightening discussions.

References

[1] P. Arrighi & A. Díaz-Caro (2012): *A System F Accounting for Scalars*. *Logical Methods in Computer Science* 8(1:11), doi:10.2168/LMCS-8(1:11)2012.

[2] P. Arrighi, A. Díaz-Caro & B. Valiron (2012): *A Type System for the Vectorial Aspects of the Linear-Algebraic Lambda-Calculus*. In E. Kashefi, J. Krivine & F. van Raamsdonk, editors: *Proceedings of DCM-2011*, *EPTCS* 88, pp. 1–15, doi:10.4204/EPTCS.88.1.

[3] P. Arrighi & G. Dowek (2008): *Linear-algebraic λ-calculus: higher-order, encodings, and confluence*. In A. Voronkov, editor: *Proceedings of RTA-2008*, *LNCS* 5117, pp. 17–31, doi:10.1007/978-3-540-70590-1_2. Available at `arXiv:quant-ph/0612199`.

[4] A. Assaf & S. Perdrix (2012): *Completeness of Algebraic CPS Simulations*. In E. Kashefi, J. Krivine & F. van Raamsdonk, editors: *Proceedings of DCM-2011*, *EPTCS* 88, pp. 16–27, doi:10.4204/EPTCS.88.2.

[5] H. Barendregt (1984): *The Lambda Calculus: Its Syntax and Semantics*. North-Holland, Amsterdam.

[6] G. Boudol (1994): *Lambda-Calculi for (Strict) Parallel Functions*. *Information and Computation* 108(1), pp. 51–127, doi:10.1006/inco.1994.1003.

[7] A. Bucciarelli, T. Ehrhard & G. Manzonetto (2012): *A Relational Semantics for Parallelism and Non-Determinism in a Functional Setting*. *Annals of Pure and Applied Logic* 163(7), pp. 918–934, doi:10.1016/j.apal.2011.09.008. Available at `hal.inria.fr:inria-00628887`.

[8] P. Buiras, A. Díaz-Caro & M. Jaskelioff (2012): *Confluence via Strong Normalisation in an Algebraic λ-Calculus with Rewriting*. In S. Ronchi della Rocca & E. Pimentel, editors: *Proceedings of LSFA-2011, EPTCS* 81, pp. 16–29, doi:10.4204/EPTCS.81.2.

[9] T. Coquand & G. Huet (1988): *The Calculus of Constructions*. *Information and Computation* 76(2–3), pp. 95–120, doi:10.1016/0890-5401(88)90005-3. Available at `hal.inria.fr:inria-00076024`.

[10] U. de'Liguoro & A. Piperno (1995): *Non Deterministic Extensions of Untyped λ-calculus*. *Information and Computation* 122(2), pp. 149–177, doi:10.1006/inco.1995.1145.

[11] M. Dezani-Ciancaglini, U. de'Liguoro & A. Piperno (1996): *Filter models for conjunctive-disjunctive lambda-calculi*. *Theoretical Computer Science* 170(1–2), pp. 83–128, doi:10.1016/S0304-3975(96)80703-1.

[12] M. Dezani-Ciancaglini, U. de'Liguoro & A. Piperno (1998): *A filter model for concurrent λ-calculus*. *SIAM Journal on Computing* 27(5), pp. 1376–1419, doi:10.1137/S0097539794275860.

[13] R. Di Cosmo (1995): *Isomorphisms of types: from λ-calculus to information retrieval and language design*. Progress in Theoretical Computer Science, Birkhauser, doi:10.1007/978-1-4612-2572-0.

[14] A. Díaz-Caro, G. Manzonetto & M. Pagani (2013): *Call-by-value non-determinism in a linear logic type discipline*. In S. Artemov & A. Nerode, editors: *Proceedings of LFCS'13, LNCS* 7734, pp. 164–178, doi:10.1007/978-3-642-35722-0_12.

[15] A. Díaz-Caro, S. Perdrix, C. Tasson & B. Valiron (2010): *Equivalence of Algebraic λ-calculi*. In: *HOR-2010*, pp. 6–11. Available at `arXiv:1005.2897v1`.

[16] A. Díaz-Caro & B. Petit (2012): *Linearity in the non-deterministic call-by-value setting*. In L. Ong & R. de Queiroz, editors: *Proceedings of WoLLIC'12*, *LNCS* 7456, pp. 216–231, doi:10.1007/978-3-642-32621-9_16. Available at `arXiv:1011.3542`.

[17] G. Dowek, T. Hardin & C. Kirchner (2003): *Theorem proving modulo*. *Journal of Automated Reasoning* 31(1), pp. 33–72, doi:10.1023/A:1027357912519.

[18] G. Dowek & T. Jiang (2011): *On the expressive power of schemes*. *Information and Computation* 209, pp. 1231–1245, doi:10.1016/j.ic.2011.06.003.

[19] G. Dowek & B. Werner (2003): *Proof normalization modulo*. *The Journal of Symbolic Logic* 68(4), pp. 1289–1316, doi:10.2178/jsl/1067620188.

[20] J. Garrigue & H. Aït-Kaci (1994): *The typed polymorphic label-selective λ-calculus*. In: *Proceedings of POPL'94*, ACM SIGPLAN, pp. 35–47, doi:10.1145/174675.174434.

[21] G. Manzonetto (2008): *Models and theories of lambda calculus*. Ph.D. thesis, Università Ca'Foscari (Venice) and Université Paris Diderot (Paris 7). Available at `tel.archives-ouvertes.fr:tel-00715207`.

[22] P. Martin-Löf (1984): *Intuitionistic type theory*. Studies in proof theory, Bibliopolis.

[23] C. Monroe, D. Meekhof, B. King, W. Itano & D. Wineland (1995): *Demonstration of a Fundamental Quantum Logic Gate*. *Physical Review Letters* 75(25), pp. 4714–4717, doi:10.1103/PhysRevLett.75.4714.

[24] M. Pagani & S. Ronchi Della Rocca (2010): *Linearity, non-determinism and solvability*. *Fundamental Informaticae* 103(1–4), pp. 173–202, doi:10.3233/FI-2010-324.

[25] B. Valiron (2010): *Orthogonality and Algebraic Lambda-Calculus*. In B. Coecke, P. Panangaden & P. Selinger, editors: *Proceedings of QPL-2010*, pp. 169–175. Available at `http://www.cs.ox.ac.uk/people/bob.coecke/QPL_proceedings.html`.

[26] L. Vaux (2009): *The algebraic lambda calculus*. *Mathematical Structures in Computer Science* 19(5), pp. 1029–1059, doi:10.1017/S0960129509990089.

[27] W.K. Wootters & W.H. Zurek (1982): *A Single Quantum Cannot be Cloned*. *Nature* 299, pp. 802–803, doi:10.1038/299802a0.